UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/528,381	03/17/2000	Anindya Basu	Basu 1-1	3850

7590 10/09/2003

Henry T. Brendzel
P O Box 574
Springfield, NJ 07081

EXAMINER	
ALI, SYED J	
ART UNIT	PAPER NUMBER
2127	5

DATE MAILED: 10/09/2003

Please find below and/or attached an Office communication concerning this application or proceeding.

PTO-90C (Rev. 10/03)

7

Office Action Summary	**Application No.** 09/528,381	**Applicant(s)** BASU ET AL.	
	Examiner Syed J Ali	**Art Unit** 2127	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *July 9, 2003* .

2a)□ This action is **FINAL**. 2b)☒ This action is non-final.

3)□ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-34* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)□ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-34* is/are rejected.

7)□ Claim(s) _____ is/are objected to.

8)□ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)□ The specification is objected to by the Examiner.

10)□ The drawing(s) filed on _____ is/are: a)□ accepted or b)□ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)□ The proposed drawing correction filed on _____ is: a)□ approved b)□ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)□ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)□ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)□ All b)□ Some * c)□ None of:

1.□ Certified copies of the priority documents have been received.

2.□ Certified copies of the priority documents have been received in Application No. _____ .

3.□ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)□ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) □ The translation of the foreign language provisional application has been received.

15)□ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) □ Interview Summary (PTO-413) Paper No(s). _____ .
2) □ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) □ Notice of Informal Patent Application (PTO-152)
3) □ Information Disclosure Statement(s) (PTO-1449) Paper No(s) _____ .	6) □ Other: .

DETAILED ACTION

1. This office action is in response to Amendment A, paper number 4, which was received July 9, 2003. Applicant's arguments have been fully considered but they deemed to be moot in view of the new ground of rejection. All previous rejections are hereby withdrawn. Claims 1-34 are presented for examination.

2. The text of those sections of Title 35, U.S. code not included in this office action can be found in a prior office action.

Claim Rejections - 35 USC § 112

3. Claims 4 and 30 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

As per claim 4, it recites the limitation "said arrangement" in line 2. There is insufficient antecedent basis for this limitation in the claim.

As per claim 30, it recites the limitation "said standalone system" in line 3. There is insufficient antecedent basis for this limitation in the claim.

Claim Rejections - 35 USC § 102

4. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

> A person shall be entitled to a patent unless –
>
> (e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

5. Claims 30-31 are rejected under 35 U.S.C. 102(e) as being anticipated by Eliott (USPN 6,468,160).

As per claim 30, Eliott discloses a storage medium that stores a control routing for use by a system to assure security of said system, the control routine including instructions for:

booting said standalone host with an authenticated operating system located on said storage medium (col. 10 lines 26-56, "serial peripheral interface 138 also includes a 'boot TOM (read only memory)' that stores a small amount of initial program load (IPL) code");

verifying an operating system of said system (col. 26 lines 10-16, "The operating system of the video game system 50 is likewise authenticated so that the presence of authentic code in both the video game system and expansion device is verified");

transferring control of said system to operating system on said system when said operating system on said system is verified (Fig. 17, wherein once the verification procedure has been completed, control shifts from the kernel to the operating system, allowing the system to run the desired applications.

As per claim 31, Eliott discloses the storage medium of claim 30, wherein said control routine verifies said operating system of said system by reading executable modules of said operating system of said system, determining a cryptographic hash for said executable modules, and comparing said cryptographic hash to a known value (Fig. 17, elements 514, 518, 520, and 528, wherein a verification procedure uses a hash value to verify the operating system).

Claim Rejections - 35 USC § 103

6. Claims 1-6, 12-15, and 25 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hanif et al. (USPN 6,141,667) (hereinafter Hanif) in view of Johansson et al. (USPN 6,466,559) (hereinafter Johansson).

As per claim 1, Hanif discloses a system including a processor, and a collection of resources interacting with said processor, said resources including at least a memory and a library of executable modules that are supported by an operating system, the improvement comprising:

a plurality of processing stacks (col. 1 lines 58-67, "To improve the processing performance of the file server, the file server is implemented as a multithreaded process", "Each thread of execution has its own stack and therefore requires more memory", wherein each thread of execution operates on system resources through its individual execution stack), each including a predefined set of at least one mediation module that processes an applied signal to form a signal that is applied to said at least one resource of said collection of resources; col. 4 lines 24-29, "Once the transaction request packet 158 is received by the socket 156 of the responding server 152, the transaction request is serviced and the responding server 152 returns a transaction response packet", wherein the applied signal is formed when the transaction request is serviced, and the socket acts as the mediation module by receiving the request, and directing the request to the processing resources); and

a service director module that intercepts requests of different types that are directed to said resources, classifies said requests in accordance with said types of said requests, and directs said requests to different ones of said processing stacks, based on said classifying (col. 4 line 30 - col. 6 line 22, "In order to open and then maintain sessions, the ASP 130 utilizes two different types of sockets for receiving two different types of requests", wherein the types of requests vary between network requests and requests to open a session, and the ASP classifies the requests and forwards it along to the execution stack of an appropriate thread).

Johansson discloses the following limitation not shown by Hanif, specifically that each different one of said resources are responsive to requests of a different type (col. 15 line 49 - col. 16 line 13, "Another aspect of the invention relates to efficient allocation of resources from different pools of resource units. A request for resources very often involves allocation of different types of resource units").

It would have been obvious to one of ordinary skill in the art to combine Hanif with Johansson since the resource allocation procedure would thus be modified such that not only can different types of requests be serviced by a socket handler, but different types of system requests can be handled differently, such as a request for hardware versus software resources. Hanif provides a way of allowing different types of sockets to either open or close a session. Thereafter, threads in a thread pool, where each thread has a separate execution stack, service network requests. However, these threads are not defined such that one thread only handles one type of request. Johansson discloses a way of allowing different data structures to service different types of system requests, such as a request for hardware resources versus a request for software resources. However, this is merely one example of the different types of resources, and the data structures could be modified to suit any number of different types of resources. To that end, the combination of Hanif and Johansson provide a method of discerning not only between front-end request types, as in different ways of acting on a socket, but also differentiating between different types of computing resources.

As per claim 2, Hanif discloses the system of claim 1 wherein said at least one resource to which said signal is applied develops an output signal that is accepted by said at least one mediation module (col. 4 lines 24-29, "the transaction request is serviced and the responding server 152 returns a transaction response packet 160 reporting the transaction outcome").

As per claim 3, Hanif discloses the system of claim 1, wherein at least one processing stack of said plurality of processing stacks comprises an ordered sequence of at least two mediation modules (col. 5 lines 21-34, "The file server software 24′ of the present invention includes a plurality of session protocol (SP) threads 202, and three queues Q1, Q2, and Q3 for maintaining AFP session requests", wherein the three queues are essentially an ordered sequence of mediation modules, in that each queue has a set of requests to be serviced on a first come first serve basis. Furthermore, in accordance with the disclosure of Johansson, each request can be applied to a different one of the threads, i.e., processing stacks, after classifying the type of request).

As per claim 4, Hanif discloses the system of claim 1, wherein said service director receives a request from an application that is active on said arrangement and applies said request to said at least one mediation module (col. 5 lines 12-34, "the present invention utilizes a multithreaded file server software process for processing AFP requests, and provides a fair-use method for assigning the threads to AFP requests to provide equal processing time for all active AFP sessions", wherein requests are serviced in a round robin fashion from the processing queues for all active sessions).

As per claim 5, Hanif discloses the system of claim 4, wherein said mediation module receives a return signal from said at least one resource of said collection of resources, processes said return signal to form a processed return signal, and sends said processed return signal to said application (col. 4 lines 24-29, "the transaction request is serviced and the responding server 152 returns a transaction response packet 160 reporting the transaction outcome").

As per claim 6, Hanif discloses the system of claim 5 wherein said at least one resource of said collection of resources sends said processed return signal via said service director (col. 4 lines 24-29, "The requesting client 152 initiates a transaction by issuing a call to the ATP 120 and supplying the parameters of the request. Once the transaction request packet 158 is received by the socket 156 of the responding server 152, the transaction request is serviced and the responding server 152 returns a transaction response packet 160 reporting the transaction outcome", wherein the ATP is the service director that controls communication between the client and server).

As per claim 12, Hanif discloses the system of claim 1, wherein said service director includes:

a service request classifier that classifies a received service request (col. 4 lines 30-56, "the ASP 130 utilizes two different types of sockets for receiving two different types of requests", wherein the ASP is the classifier that determines the type of request and services it accordingly); and

Johansson discloses a processing stack selector that selects a processing stack based upon said classification, and communicates said service request to said selected processing stack (col. 16 lines 35-65, "The resource handler 200 includes a mapper 206 which establishes a relationship between ones of the resource units in the first pool mirrored by data structure 202 and resource units in the second pool mirrored by data structure 204").

As per claim 13, Hanif discloses the system of claim 1, wherein said service director includes a service request classifier that classifies a service request based upon the type of service request and arguments of the service request (col. 4 lines 24-29, "The requesting client 152 initiates a transaction by issuing a call to the ATP 120 and supplying the parameters of the request", wherein the parameters are the equivalent of the arguments and the request is serviced accordingly).

As per claim 14, Hanif discloses the system of claim 1 further comprising a connection to a network (Fig. 1, wherein a network topology is disclosed, where clients and servers may communicate via a network connection to service requests).

As per claim 15, the modified Hanif does not specifically disclose the system of claim 14 wherein said connection is secure.

"Official Notice" is taken that providing a secure network connection in order to provide network security is well known and expected in the art. It would have been obvious to one of ordinary skill in the art to provide a secure connection since protection of the connection would shut off an additional avenue of attack for an interloper. Since secure connections are well known in the art, if an effort is to be made to provide a secure system, providing a secure connection would have been an obvious modification since it effectively prevents certain types of security breaches.

As per claim 25, "Official Notice" is taken that while Hanif does not specifically disclose the system of claim 1 wherein said service director and said processing stacks are embedded in a loadable library of C language executable modules, to do so would have been obvious to one of ordinary skill in the art. That is, Hanif does not specifically state what programming language is to be used to implement multithreading, but does require that the system be compatible with the AppleTalk networking system. It is well known that the C language can provide executable code compatible with Apple systems, and since threads can be implemented in any number of programming languages, including C, C++, and Java, inter alia, it would have been an obvious modification over the disclosure of Hanif.

7. Claims 7-8, 10-11, and 23-24 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hanif in view of Johansson in view of Hershey et al. (USPN 5,414,833) (hereinafter Hershey).

As per claim 7, Hershey discloses the following limitations not shown by the modified Hanif, specifically the system of claim 1, wherein said at least one mediation module is based upon a chosen security policy (col. 23 lines 33-65, "In this security application, it is assumed that encryption is performed at the application layer", wherein it is discussed above that an application or software module could be construed as a mediation module).

It would have been obvious to one of ordinary skill in the art to combine the modified Hanif with Hershey since many security breaches involve corruption of the processing stack. Some examples of known means of breaching the stack include stack override, buffer overflow, smashing the stack, trashing the stack, mangling the stack and others. By implementing a security policy within the processing stack that can handle these types of breaches, a system can be ensured to be more secure. Since the security policy is disclosed as a software application, it may be modularized, and defined as a method within a particular thread. To that end, a specific security policy for that the type of resource that thread acts upon can be defined, allowing greater specialization in the security of the system.

As per claim 8, Hershey discloses the system of claim 1, wherein said at least one mediation module in said processing stack performs encryption (col. 23 lines 33-65, "In this security application, it is assumed that encryption is performed at the application layer", wherein it is discussed above that an application or software module could be construed as a mediation module).

As per claim 10, Hershey discloses the system of claim 1, wherein said mediation module performs authentication (col. 17 lines 8-56, "Upon detecting that program latch 302 has been set, ...processor 305 constructs a security alert message from information stored in non-volatile registers 303 and causes a message authentication code to be calculated on the security alert message by invoking data encryption algorithm 304").

As per claim 11, the modified Hanif does not specifically disclose the system of claim 1 wherein said mediation module is a secure file system.

"Official Notice" is taken that the implementation of secure file systems is well known in the art. Furthermore, since a mediation module is in essence any software module that acts as a go-between for a client and system resources and since the implementation of a secure file system is well known in the art, it would have been obvious to one of ordinary skill in the art to include such a software module in the modified Hanif for the purpose of designing a system that is resistant to security breaches. Furthermore, although Hershey does not specifically discuss secure file systems, Hershey does discuss network security, which is very much related to providing a secure file system.

As per claim 23, Hershey discloses the system of claim 1, wherein said at least one mediation module includes at least one authentication code retriever that retrieves an authentication code and a validation system that validates said service request against said authentication code (col. 17 lines 8-56, "Security alert message transmission means 306 causes the security alert message and message authentication code to be transmitted via bit stream 124 to a destination device such as a network security manager device", wherein the network security manager device would use the encryption algorithm to validate the authentication code).

As per claim 24, Hershey discloses the system of claim 1 wherein said operating system includes means to prevent implication of an operating system breach from an administrative user breach. This is done by implementation of the chosen security policy, such that if the authentication step is breached, access to the processing stacks is still controlled by the service director. Thus, administrative user breaches would be inherently prevented. That is, since all service requests go through the director, and applications may run within the sandbox without having access to change system resources, modification of the operating system would be impossible in spite of an administrative user breach, provided that the chosen security policy protects the operating system.

8. Claim 9 is rejected under 35 U.S.C. 103(a) as being unpatentable over Hanif in view of Johansson in view of Traversat et al. (USPN 6,052,720) (hereinafter Traversat).

As per claim 9, Traversat discloses the following limitations not shown by the modified Hanif, specifically the system of claim 1, wherein said mediation module is a namespace manager (col. 7 lines 31-36, "The namespace manager controls how the entries are stored and accessed within the namespace. The manager implements a standard interface that exports the security, storage, and ownership attributes of any entry in the namespace").

It would have been obvious to one of ordinary skill in the art to combine the modified Hanif with Traversat since Traversat provides a way of easily routing service requests to the appropriate processing stack. For instance, each particular processing stack may be responsible for performing certain types of tasks. The parameters passed by the application to the service director disclosed by Hanif (col. 4 lines 24-29, "The requesting client 152 initiates a transaction by issuing a call to the ATP 120 and supplying the parameters of the request") could contain the destination namespace, and routing of the request to the appropriate stack would be as simple as matching the namespaces..

9. Claims 16-17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hanif in view of Johansson in view of Pierce, Jr. et al. (USPN 6,560,217) (hereinafter Pierce).

As per claim 16, Pierce discloses the following limitations not shown by the modified Hanif, specifically the system of claim 14, wherein said network is a virtual private network (col. 8 lines 18-39, "Each home agent is associates with one of the virtual private networks, and each home agent has or is associated with a unique IP address").

It would have been obvious to one of ordinary skill in the art to combine the modified Hanif with Pierce since it would provide a way of allowing the network interface disclosed by the combination of Hanif and Johansson in claims 1 and 14 to be implemented in such a way as to provide both the scalability and functionality of a public network, by allowing client-server communication, while also allowing the user the management tools associated with a private network, such as supporting data routing for each processing stack individually.

As per claim 17, "Official Notice" is taken that to provide the system of claim 16 wherein said connection is secured would have been obvious to one of ordinary skill in the art. This is discussed above in reference to claim 15. The discussion of that claim also discusses the added benefit that could be gained by providing a secure network connection, such as preventing security breaches through stack corruption.

10. Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over Hanif in view of Johansson in view of Pierce as applied to claims 1, 14, and 16 above and further in view of Hershey.

As per claim 18, Hershey discloses the system of claim 17 wherein said connection is secured through encryption (col. 23 lines 33-65, "In this security application, it is assumed that encryption is performed at the application layer").

11. Claims 19-22 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hanif in view of Johansson in view of Bond et al. (USPN 6,275,938) (hereinafter Bond).

As per claim 19, Bond discloses the following limitations not shown by the modified Hanif, specifically the system of claim 1 further comprising a compliance supervisor that is coupled to said processing stacks, and to said service director, and is adapted for receiving security policy information from outside said system (col. 8 lines 12-27, "The location of WHKRNL32 352 outside sandbox 215 is especially important, because it is here that the security policy is actually implemented", wherein Bond discloses a module located outside a sandbox that implements a security policy for that sandbox).

It would have been obvious to one of ordinary skill in the art to combine the modified Hanif with Bond since it provides an added security benefit to implement the security policy outside of the processing stack. This would prevent an application from modifying the security policy if it were to gain access to the stack, thus cutting off an avenue of attack.

As per claim 20, Bond discloses the system of claim 19, wherein said compliance supervisor receives said security policy information from a virtual private network (col. 1

lines 24-34, “The platform-independent tokenized byte code runs on a virtual machine which places strict limits on what the executable code can do”, wherein Bond discloses prior art that shows that to implement particular security policies, particularly in conjunction with sandboxes, is well known to also implement those policies on virtual systems).

As per claim 21, Bond discloses the system of claim 19, wherein said compliance supervisor includes a processing stack modifier that modifies said processing stack based upon a received security policy (col. 7 lines 51-64, “Wx86VM loads API thunk DLLs (secure APIs) such as 391 into the sandbox”, wherein the virtual machine loads the security policy from WHKRNL32 into the sandbox, thereby modifying the processing stack in such a way as to adhere to that policy in handling function and memory calls).

As per claim 22, Hanif discloses the system of claim 19, wherein said compliance supervisor includes a processing stack creator that creates a processing stack based upon said security policy (col. 5 lines 28-34, “The file server software 24′ of the present invention includes a plurality of session protocol (SP) threads 202”, wherein the creation of a new thread would lead to the creation of a new processing stack. Furthermore, since the virtual machine is controlled by the security policy, whatever processing stacks are created must adhere to that policy).

12. Claims 26-29 are rejected under 35 U.S.C. 103(a) as being unpatentable over Hanif in view of Johansson as applied to claim 1 above, and further in view of Eliott.

As per claim 26, Eliott discloses the following limitations not shown by the modified Hanif, specifically the system of claim 1 further comprising a read-only program store that is read by said system upon boot-up (col. 10 lines 26-56, "serial peripheral interface 138 also includes a 'boot TOM (read only memory)' that stores a small amount of initial program load (IPL) code").

It would have been obvious to one of ordinary skill in the art to combine the modified Hanif with Eliott for the purpose of ensuring that the operating system could not be altered during boot-up. By making the program read-only, the operating system could not be breached, and thus the system would become more secure. This also prevents any intruder from installing various backdoors for entry at a later point, such as a cuckoo's egg or a Trojan horse. By requiring that that the boot-up system is read-only, a common avenue of attack is shut off.

As per claim 27, Eliott discloses the system of claim 26, wherein said system includes an operating system, and said read-only program store contains a program module for verifying the operating system, and authentication program modules for authenticating software present in said memory of said system (col. 26 lines 10-16, "The operating system of the video game system 50 is likewise authenticated so that the presence of authentic code in both the video game system and expansion device is verified").

As per claim 28, Eliott discloses the system of claim 27 where said software that is authenticated by said authentication program modules includes software that forms an operating system of said system (col. 26 lines 10-16, "Resident in boot ROM 182 is a set of instructions which permit the remainder of the expansion device operating system to be accessed", wherein the combination of the boot ROM and the expansion device boot system form the operating system for the video game system).

As per claim 29, Eliott discloses the system of claim 28 where said authentication program modules develop a cryptographic hash of software to be authenticated (col. 26 lines 17-21, "Any of various available encryption algorithms may be utilized in order to obtain the desired degree of security", wherein all software on the system is to be authentication using whatever encryption algorithm is chosen to be used in the system).

13. Claims 32-34 are rejected under 35 U.S.C. 103(a) as being unpatentable over Eliott in view of Hanif in view of Johansson.

As per claim 32, it is rejected for similar reasons as stated above for claim 1. Specifically, the combination of a plurality of processing stacks and a service director make up the reverse sandbox. Therefore, the combination discussed in claim 1, of intercepting requests and directing the request to the appropriate resource meets this limitation. Furthermore, the discussion of claim 29 shows that all software being loaded on the system of Eliott must be authenticated using whatever encryption algorithm is implemented on the system. To that end, the reverse sandbox would have to be verified against said algorithm before control could be transferred to it. It would have been obvious to one of ordinary skill in the art to combine Eliott with Hanif and Johansson for purposes of providing a verified operating system, in accordance with claim 30, while also protecting data from intruders by not allowing users access to the processing stacks. Ensuring that all service requests are verified against a security policy, and requiring all access to be originated from outside the system can protect resources to further prevent system breaches.

As per claims 33 and 34, they are rejected for similar reasons as stated above. Any software module, such as a reverse sandbox, would have to be installed on a system in order for it to function in conjunction with the operating system. Furthermore, the service director, compliance modules, processing stacks, and mediation modules are all discussed above thoroughly.

Response to Arguments

14. Applicant argues that the previously cited Bak and Brandle references do not teach the limitations of amended claim 1, especially in reference to having "**a predefined set of at least one mediation module.**" Examiner agrees with this argument, and has therefore withdrawn the rejection in question. Newly cited references Hanif and Johansson, however, do disclose processing stacks that have a predefined set of at least one mediation module, and also that the different processing resources are usable for servicing different types of requests. That is, Johansson discloses a way of defining separate data structures for separate types of system resources, and directing a request to

the appropriate data structure when a request is received. Similarly, Hanif discloses a way of receiving different types of requests and classifying those requests accordingly. Although Hanif is related to servicing different types of socket requests, the idea can be expanded to also consider different types of system requests, such as hardware versus software requests. Further, since Hanif discloses a multithreaded system, where each thread has its own processing stack, the combination of this type of data structure with the disclosure of Johansson of allowing different data structures for different types of requests would result in a system containing all the limitations of claim 1. Furthermore, the motivation for combining, as discussed above in reference to claim 1, exists in the sense that the system not only protects resources from intruders by preventing access to the stacks, but also allows the designer of the system more flexibility in that more specialized thread methods can be defined that are only operable for one type of resource, thereby adding another degree of functionality.

15. Applicant argues on page 11, "*the monitor of Hershey et al is not a module in a stack. Indeed, it appears that the monitor of Hershey et al is not a software module but, rather, a set of parallel hardware finite state machines.*" Examiner respectfully disagrees. Hershey discloses how a security policy, implemented as a security application, can be used to verify a bit stream. Clearly, since the security policy is implemented as an application, it should be construed as a software module. While it is acknowledged that the security agent is used to verify encrypted bit streams that may be transmitted at a hardware level, the interpretation of the bit stream and the security messages generated therein must be done in software. Furthermore, it is not the

contention the Examiner that the security module or the security alert messages of Hershey is implemented in a stack. Rather, it is felt that such a security policy could be defined within the processing stack of any of the plurality of threads defined by the modified Hanif. Depending on the security policy necessary for the particular sets of resources, security policies could be chosen to suit those particular needs and provide a more robust system. Any number of private methods could be defined for each thread, corresponding to very specific security policies, and able to send and receive alert messages related to specific types of security faults. This also relates to Applicant's argument that "*the mere knowledge that secure file systems can be created adds little to the combination of Bak, Brandle et al, and Hershey et al.*" However, since the combination of Hanif, Johansson, and Hershey is related to the protection of different types of system resources, as well as providing security against intrusions, the provision of a secure file system would have been an obvious modification. That is, in order to provide the most reliable system security, all avenues of attack must be accounted for. Since secure file systems are a very well known way of preventing unauthorized access, it would have been obvious to include this as an added layer of security.

16. Applicant argues on page 12 that "*Means 306 is not a module on a stack, there is no teaching that it* ***retrieves*** *an authentication code, and means 306 does not serve as a module that validates a service request against the authentication code.*" Regarding the first argument, this is a similar argument as the one presented above in reference to heading 15. Applicant also presents a similar argument on page 12 to overcome the rejection of Claim 9 over Bak in view of Brandle et al. in view of Traversat. That is, it

would have been obvious to implement the security alert message or the namespace manager within a module in the stacks, and the citation from the reference should be viewed as how it would be combined with Hanif and Johansson. Furthermore, concerning retrieving an authentication code, this is actually performed by the retrieval of information from the non-volatile registers, which then results in the calculation of the authentication code. While the code is not retrieved directly, the code is applied from values retrieved from the registers.

17. Applicant's argument regarding the previous rejection of claim 16 does not apply in view of the new grounds of rejection. Specifically, Hanif is disclosed in a networking environment, connected to another network via a plurality of routers. Therefore, the scope is not limited simply to a local area network, but to any other computer or network that can be reached by that local area network.

18. Applicant argues on page 13 that "*it still remains that claim 19 specifies that the compliance supervisor is one that is adapted for receiving security information from outside the system, and there is no indication in Bond et al. that module WHKRNL32 is modifiable. If it is not modifiable, then it is NOT adapted 'for receiving security policy information from outside said system,' as claim 19 specifies.*" However, Bond does state that WHKRNL32 is in fact modifiable. In particular, the fact that it is modifiable and the security policy can be changed is exactly the reason for its location outside the sandbox (col. 8 lines 12-27, "if it were inside the sandbox, a rogue applet might be able to compromise security by modifying it"). Furthermore, the module WHKRNL32 is

coupled to the processing stacks and service director, in that it provides communication with the sandbox. As these elements have been shown within Hanif and Johansson as well, the module therein defining the security policy would be coupled to the service director and processing stacks via the combination with Hanif and Johansson.

Further, Applicant argues on page 14 that "*the security policy…is not carried out by the compliance supervisor but, rather, by the mediation modules.*" In fact, this is also what Bond discloses. While the security policy is implemented within the module WHKRNL32, it is not actually carried out by this module, but rather it is held there. Specifically, the API DLLs load information from outside the sandbox, and remap those policies to DLLs within the sandbox so that they can be used by the system (col. 8 lines 51-64, "Wx86Vm loads API thunk DLLs (secure APIs) such as 391 into the sandbox. Wx86VM is able to modify the names of DLLs within the operating-system loader").

19. The remainder of the arguments presented allege that dependent claims are allowable for at least the same reasons as their parent claims. These arguments have been addressed above, and therefore the claims stand rejected in view of the above arguments, as well as the new grounds of rejection.

Conclusion

20. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Syed J Ali whose telephone number is (703) 305-8106. The examiner can normally be reached on Mon-Fri 8-5:30, 2nd Friday off.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, William A Grant can be reached on (703) 308-1108. The fax phone number for the organization where this application or proceeding is assigned is (703) 872-9306.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

Syed Ali
September 22, 2003

MAJID A. BANANKHAH
PRIMARY EXAMINER